Git Guide

Git is officially and basically a *distributed version control system (VSC).* It is a technology that used by different providers like GitHub, GitLab and so on.

Setup

In order to setup (*https://git-scm.com/downloads*) Git onto Ubuntu,

```
$  sudo apt-get install git
```

To control both the installment and the version,

```
$  git --version
```

Commands

This name and email are shown in the logs of comments so it has nothing to do with any connection to any services but in terms of clearness it is strongly recommended.

```
$  git config --global user.name "Name"
$  git config --global user.email "Mail"
```

To initialize the local folder as Git location,

```
$  git init
```

To check the current status of the files,

```
$  git status
```

There is a place called 'staging area' where we should add our modified files so that we can make a commit. To move the wanted files to the staging area,

```
$  git add <file_name.extension>
$  git add <file1_name.extension> <file2_name.extension>
```

or to move all the files,

```
$  git add .
```

So after moving things to staging area we can now comment,

```
$  git commit -m "Commit Message"
```

to check the comment logs and view their hash numbers,

```
$  git log
$  git log --oneline
```

In order to switch to a branch, which is generally the main branch, master,

```
$  git checkout <branch-name>
$  git checkout master
```

or to the specific comment, this will let us to see the situation of the project at that specific date and time and not delete or ruin anything. It's like moving in a time machine...

```
$  git checkout <commit-bash-wanted-to-switched>
```

To list all the branches,

```
$  git branch
```

To create a new branch,

```
$  git branch <branch-name-to-be-created>
```

Both to create and switch to a new branch at the same time,

```
$  git checkout -b <branch-name-to-be-created>
```

To merge a branch with the current one,

```
$  git merge <branch-name-to-be-merged-into-current-branch>
```

To revert a specific commit, which undoes just that specific comment, and adds a new comment like 'Reverted x comment',

```
$  git revert <commit-bash-to-be-undone>
```

To reset back to a specific comment, which deletes and destroys every other thing after that specific comment, but preserves the deleted things on our editor just in case, if we want to readd somethings and recomment them again etc.

```
$  git reset <commit-bash-wanted-to-reset-back-to>
```

But if we use it with "--hard" tag at the end, it will delete also those things on our editor completely and will be no turning back from that point,

```
$  git reset <commit-bash-wanted-to-reset-back-to> --hard
```

To delete a branch completely,

```
$  git branch -d <branch-name-to-be-deleted>
```

GitHub

Until here all the operations were made locally, in order to send them online and collaborate wtih others we need to do the followings. First we have to give our remote git address an alias name, which is generally used as 'origin',

```
$  git remote add <alias-name> https://github.com/<uname>/<repname>.git
$  git remote add origin https://github.com/<username>/<reponame>.git
```

After working on things and moving the needed files to the staging area and making comments, to deploy the changes and so the comments to the remote Git server, so here we can decide to which branch we want to deploy the stuff,

```
$  git push -u <alias-name> <branch-name>
$  git push -u origin master
```

To copy the changes on the remote Git repository to the local folder,

```
$  git pull <alias-name> <branch-name>
$  git pull origin master
```

To clone a repository, which is generally a fork one,

```
$  git clone <github-repo-link>
```

Ignoring Things

In order to ignore any files or folders and prevent them being tracked by Git, a file named as "*.gitignore*" can be created. Inside this file, all the things to be ignored must be written line by line.

Resources & Links

https://www.youtube.com/watch?v=USjZcfj8yxE
https://www.notion.so/Introduction-to-Git-ac396a0697704709a12b6a0e545db049
https://www.youtube.com/watch?v=nhNq2kIvi9s
https://www.notion.so/Introduction-to-GitHub-202af6f64bbd4299b15f238dcd09d2a7
https://www.youtube.com/watch?v=RIYrfkZjWmA